Un enseignement au numérique pour notre académie : Les algorithmes

De l'algorithme d'Euclide à l'algorithmique

Marie-Danièle Campion, recteur de l'académie de Clermont-Ferrand

Les programmes informatiques tels qu'ils apparaissent directement à l'écran dans des circonstances de panne informatique renvoient chacun d'entre nous à un vide culturel. L'expression langage informatique est d'ailleurs couramment utilisée pour qualifier un langage accessible aux ordinateurs et qui dépasse la plupart du temps l'utilisateur dans ses aspects conceptuels. Peut-être pouvons-nous a priori penser que les algorithmes sont étroitement liés au monde de l'informatique et pour éviter cet écueil, il est important d'avoir des notions précises sur ce que l'on appelle un algorithme.

Cependant bien qu'il faille à la base dissocier algorithmique et informatique, l'exécution de certains algorithmes est de plus en plus confiée à l'ordinateur du fait que cet outil technologique supplante l'humain en termes de vitesse d'exécution, de capacité de mémoire et de qualité du traitement du signal. En ce sens, l'algorithmique est aujourd'hui une discipline très orientée vers le développement de logiciels informatiques qui mettent en œuvre des algorithmes pensés et réfléchis.

Avec ce document, nous souhaitons d'une part décortiquer cette notion d'algorithme et d'autre part mettre en avant le cheminement intellectuel conduisant de la formulation d'un ensemble de tâches à effectuer jusqu'au développement d'un programme exécutable sur ordinateur. C'est indirectement un double objectif que nous avons ici pour les élèves : leur montrer que les algorithmes font partie de notre quotidien et leur faire percevoir la notion de programmation informatique. Je vous souhaite ici une bonne et fructueuse lecture.

La notion d'algorithme au cœur des enseignements du secondaire

Karim Zayana, inspecteur général de l'éducation nationale

Qu'est-ce qu'un algorithme ?

D'abord d'où vient ce terme. Certainement pas du mot « rythme », qui s'orthographie avec un « y », mais du mathématicien perse El Khawarizmi (IXème siècle). L'article défini « El », qui signifie « le », s'infléchit en français en « al », et le « Kh » se prononce en vérité comme la jota espagnole, ici approximativement rendue par un

« g » guttural. Le nom de ce savant mort à Bagdad s'est alors occidentalisé en « algorithme », tout simplement car c'est à peu près ainsi qu'il sonne en arabe.

Venons-en à l'objet conceptuel qui se cache derrière ce mot.

De façon très générale, un algorithme désigne un schéma de pensée dictant une séquence non ambigüe de tâches à réaliser. Une recette de cuisine est un algorithme, certes simple.

Pour autant, un algorithme n'est pas un long fleuve tranquille. On peut y croiser :

- *des tests*, conduisant à des choix. En cuisine, une recette peut être déclinée en deux versions, selon qu'on est équipé du gaz ou de l'électrique. Sur un ordinateur, l'alternative est rendue par des instructions du type : « if ... { } else { } », et le test d'égalité, souvent, par un = redoublé, à savoir un « = = ».
- *des tâches répétitives*, dont le nombre de rondes est parfois prévisible, parfois pas. En cuisine, cela peut correspondre aux consignes « attendre 10 minutes, c'est-à-dire faire l'instruction RIEN pendant 1 minute, puis faire l'instruction RIEN pendant 1 minute, et la même chose encore 8 fois de suite » (boucle conditionnelle) ou bien « attendre, c'est-à-dire faire l'instruction RIEN, jusqu'à ce que l'eau bout » (boucle inconditionnelle). On fait bégayer l'ordinateur grâce à ces commandes, facilement reconnaissables : « for( ) { }» ou bien « while( ) { } », un peu comme s'il s'agissait d'une punition qu'on lui infligeait.
- *des interactions avec l'environnement*, ce qu'on appelle aussi des entrées et sorties, par l'intermédiaire de variables qui seront initialisées et/ou modifiées en cours de route. En cuisine, on agit sur le minuteur qu'on initialise à 10 minutes, on utilise des ingrédients, on les transforme pour produire quelque chose. Sur un ordinateur, l'affectation est souvent traduite par un « := », ou un « = », à ne pas confondre avec le « = = » qui indique un test.
- *des sous-tâches*, elles-mêmes éventuellement fractionnables. C'est le principe de la modularité, à savoir la décomposition d'un programme en sous-programmes, fonctions, procédures, méthodes. En cuisine, un commis sera spécialisé dans la réalisation des sauces, un autre des soupes, un autre encore des cocktails, etc. Le chef les sollicitera autant de fois que cela sera nécessaire pour réaliser notre mets préféré.

Pourquoi en faire en mathématiques ?

Parce que les langages mathématiques et informatiques s'éclairent mutuellement :
Dans les deux champs, il y est par exemple question de fonctions, de variables, de paramètres, de conditions et d'hypothèses, de logique, de tables indexées (nos suites en mathématiques), de boucles (de sommation par exemple, en mathématiques), de récursivité (nos récurrences, en mathématiques). Autant de *concepts voisins* qui trouvent ainsi des débouchés très parlants.

La mise en œuvre d'un algorithme sur machine interroge sur la *représentation des nombres*, les approximations de calculs, le coût de leur réalisation. L'émergence de ces questions appelle à son tour des réponses d'ordre mathématique.

Certaines *preuves ou conjectures* sont aujourd'hui établies à l'aide de l'ordinateur, dont on exploite alors les capacités de calculs et les possibilités graphiques.

Avec quoi commencer ?

Au collège ou même avant le collège, on peut recommander de commencer avec un ordinateur sur lequel on aura installé le logiciel éducatif Scratch. Développé par le MIT et en téléchargement libre, Scratch initie de façon ludique, graphique et très stimulante à la programmation (y compris à une pensée orientée objet) en passant en revue les fondamentaux du genre et en les mettant à la portée d'un enfant.

Avec quels outils en faire par la suite ?

On peut constater qu'au lycée plusieurs élèves sont déjà capables de passer à la vitesse supérieure. Pour ceux-là, prévoir un ordinateur, et, quand cela est possible, une interface matérielle dotée de capteurs (température, pression, lumière, son, etc.) et d'actionneurs (moteur, diode, buzzer, etc.) qui seront les images palpables de nos variables. Cela rendra d'autant plus vivante et interdisciplinaire l'activité. Il existe aujourd'hui :

- des environnement de développement très ergonomiques, librement téléchargeables, et non artificiels, c'est-à-dire non réservés au seul public scolaire, permettant de programmer dans des langages largement répandus, polyvalents, abordables avec un niveau intermédiaire et permettant une forte progression : C, Python, Scilab pour ne citer qu'eux.
- des cartes à microcontrôleurs, type Arduino, très bon marché (15 €) et du petit outillage électronique qui l'est tout autant (quelques cents le composant, deux ou trois euros le petit moteur, etc.).

Les vertus de l'interdisciplinarité

En conclusion, insistons sur ce point, qui rend plus concret les concepts étudiés à travers, par exemple, la réalisation de maquettes.

Pensez aux conjonctions logiques, le « ou » et le « et ». Pour les introduire, on peut prendre le sempiternel exemple du « fromage et/ou dessert » qui : 1) n'est pas un test 2) oblige à distinguer le « ou » littéral (exclusif dans le sens de la phrase) du « ou » logique (inclusif). Une autre façon plus performante de les introduire consiste à dire ceci :

- le « ou » logique fonctionne comme une alarme (n'importe lequel des boutons, câblés en parallèle, peut déclencher l'alarme en faisant contact).
- le « et » logique fonctionne comme sur une machine sécurisée (une presse d'emboutissage, un taille-haie) : il faut appuyer simultanément sur deux boutons (câblés en série) assez distants pour assurer qu'on n'y laisse aucune main.

C'est ainsi que, lorsqu'elles sont incarnées, ces conjonctions logiques sont rendues beaucoup plus accessibles pour les élèves.

Algorithmes

Maurice Nivat, professeur honoraire à l'Université Paris Diderot.

Les algorithmes sont très anciens, aussi anciens que l'*homo sapiens :* nos ancêtres préhistoriques en utilisaient à coup sûr ; dès que l'on veut faire quelque chose qui n'est pas évident, qui implique plusieurs actions simples, on est obligé d'utiliser un algorithme qui est la planification de la succession de ces opérations simples qu'elles soient réalisées par un seul acteur ou plusieurs, que certaines opérations puissent être effectués simultanément ou non. Les algorithmes qui permettaient de chasser le mammouth avec succès, de dresser un menhir de cent tonnes, de débiter un morceau de silex en lames de rasoir sont perdus, mais on ne peut douter qu'ils aient existé.

Les algorithmes sont aussi très répandus : nous en utilisons dans notre vie quotidienne à chaque instant, la plupart du temps sans nous en rendre compte. Nous nous habillons le matin sans réfléchir, selon pourtant un algorithme que nous utilisons si souvent qu'il est devenu réflexe, mais il suffit d'un incident pour que l'on doive changer d'algorithme, l'entorse que je me suis faite hier à la cheville m'empêche de m'habiller comme j'en ai l'habitude, et m'oblige sans doute à mettre d'autres vêtements et d'une autre façon, une grève m'empêche de me rendre au lycée comme j'en ai l'habitude, avec le bus, que je prends machinalement, je suis obligé de trouver un autre moyen de me déplacer.
Tout ce qui est recette, méthode, mode d'emploi est un algorithme, et nous avons câblés dans notre cerveau quantité d'autres algorithmes appris (car nous ne les avions pas à la naissance) comme ceux que nous mettons en œuvre pour marcher, manger, lire, écrire, communiquer avec nos semblables, etc.

Le grand succès de l'informatique qui a envahi notre vie en soixante ans d'existence provient pour une large part du réexamen d'un très grand nombre d'algorithmes liés aux divers types de métiers ou d'activités à la lumière de leur mécanisation totale ou partielle, celle-ci consistant à confier à des machines électroniques (universelles tels les ordinateurs) ou dédiées et se résumant parfois à une simple puce. C'est cet examen qui a conduit à transformer en profondeur nombre de ces métiers et de ces activités. Cette transformation se poursuit inexorablement au fur et à mesure que la technique fait progresser les performances des machines et que la science fait progresser notre connaissance des algorithmes et des possibilités de dialogue homme-machine.
Les algorithmes une fois écrits deviennent des programmes, ils peuvent être écrits dans toutes espèces de langages, il en existe désormais des milliers qui sont soit des langages complètement artificiels qui sont dits langages de programmation, soit des portions de langues naturelles enrichies de notations des opérations simples en lesquelles l'algorithme décompose l'opération complexe qu'il est censé réaliser.
Il faut bien voir que simple ici ne veut rien dire, le jeu des opérations considérées comme simples peut en contenir certaines qui sont déjà fort compliquées. Ces

opérations compliquées sont alors réalisées par d'autres algorithmes connus et programmés de façon, précisément à pouvoir être utilisés comme opération simple (ou brique de base) dans d'autres algorithmes : la programmation de la recherche algorithmique de la solution d'un problème se fait ainsi à divers niveaux de granularité, dépendant du jeu d'opérations dites simples que l'on a à sa disposition. Tout se passe comme pour franchir un mur : si l'on a une échelle assez longue pour atteindre le haut du mur c'est assez simple, sinon cela peut être fort compliqué.

De bien inutiles et vaines polémiques agitent chercheurs et pédagogues sur ce qu'il conviendrait d'enseigner d'algorithmique aux enfants d'âge scolaire, pour certains la programmation est trop abstraite pour être enseignée avant les plus grandes classes des lycées, pour d'autres elle est une activité aussi fondamentale que lire écrire et compter et son apprentissage doit commencer en classe maternelle. Ce qu'il faut voir c'est que savoir lire écrire et compter, les trois apprentissages considérés aujourd'hui par tout le monde comme les plus fondamentaux, comporte une large part d'algorithmique et programmation : c'est un algorithme qui n'a rien de simple qui établit la correspondance entre syllabes et phonèmes qui permet de lire, c'est un algorithme qui permet de faire correspondre une suite de lettres à la suite de phonèmes qui composent une phrase ordinaire de la langue parlée, c'est un algorithme qui, avant toute arithmétique, permet de compter et de comprendre ce qu'est un nombre entier et de construire la suite des noms de nombres qui en français comme dans toute langue vernaculaire permet de désigner chaque entier et donc d'annoncer le résultat d'un comptage.

Les occasions de faire comprendre aux enfants ce qu'est un algorithme sont ainsi multiples dès le début de l'école primaire et devraient être saisies et exploitées ne serait-ce que pour leur faire comprendre la difficulté que présente chacun de ces savoirs de base et les introduire à la pensée algorithmique. L'algorithmique est elle-même plus une méthode, une façon de penser, qu'un corpus de savoirs. L'exemple du comptage est très éloquent : pour compter quoi que ce soit il faut d'abord s'assurer que les éléments à compter sont identifiables un par un et définir précisément l'ensemble des éléments à compter, faute de quoi l'opération de comptage n'a pas de sens.

Puis il faut se saisir, par un moyen ou par un autre, de chaque élément de l'espace des éléments à compter et simultanément ajouter 1 au nombre des éléments déjà comptés et faire passer l'élément saisi de l'espace des éléments à compter dans l'espace des éléments déjà comptés. On s'arrête quand l'espace des éléments à compter est vide, c'est à dire ne contient plus d'élément.

Ceci n'a pas l'air très compliqué mais de fait c'est ce que l'on apprend aujourd'hui à faire aux enfants, avec des jetons, des boites, des billes, des allumettes ou des points des croix, des cercles, des allumettes dessinés sur des pages de leur cahier d'exercice et c'est ce que font les scrutateurs à la fin d'une journée de votes pour compter les bulletins qui sont dans l'urne.

Ce que nous venons d'écrire n'est pas tout à fait un algorithme car nous n'avons pas dit comment on se saisit des éléments les uns après les autres ni comment on fait passer un élément d'un espace dans l'autre, et ceci est volontaire parce que cela va

dépendre essentiellement de la nature, de la taille, de la disposition des éléments à compter. Si ce sont des billes dans un sac, on les prend une à une et en incrémentant de 1 le nombre des éléments déjà compter (qui peut être affiché sur un compteur) on fait passer une bille après l'autre toutes les billes du sac initial dans un deuxième sac, c'est à dire on répète l'opération « prendre une bille dans le premier sac, ajouter 1 au compteur, mettre la bille dans le deuxième sac » jusqu'à ce que le premier sac soit vide. Et là nous avons un beau petit programme avec une boucle régie par un test, le test de vacuité du premier sac.
L'intérêt de notre premier pseudo programme et que s'il s'agit de compter les croix dessinées à l'intérieur d'un rectangle sur le cahier, on peut encore l'utiliser mais on ne peut pas « prendre » une croix dessinée, l'enlever du rectangle et la mettre dans un autre (à moins que les croix dessinées au crayon à mine puissent être efficacement gommées, auquel cas chaque croix est gommée en même temps qu'elle est comptée) et il faut procéder autrement, en général on le fait par marquage, chaque fois qu'on compte une croix on l'entoure d'un petit cercle qui contient cette croix et elle seule, et on poursuit jusqu'à ce que toutes les croix soient marquées ou cela revient au même qu'il n'y ait plus de croix non marquée.

Et maintenant avec un peu d'imagination on peut compter tout ce qu'on veut ou se rendre compte que l'on n'y arrivera pas : on compte les moutons dans un pré (attention les moutons bougent tout le temps) les dîneurs un restaurant (attention on ne peut pas leur demander de se déplacer comme on forcera les moutons à faire) et on renonce vite à compter les feuilles d'un arbre, non que leur nombre soit trop grand mais parce que l'on n'arrive pas à séparer les feuilles déjà comptées des feuilles restant à compter. On peut aussi s'intéresser à tous les compteurs ou machines à compter, pas forcément électroniques et dieu sait s'ils sont nombreux, compteurs de pièces et billets de banque, compteurs des véhicules qui passent en un certain point de la route, compteurs volumétriques à eau ou à gaz. Compter est la première opération de mesure d'une quantité que les enfants apprennent mais les opérations de mesure de longueur, de surface, de volume, de température, de temps ou de danger, de difficulté, de pauvreté, de richesse, de beauté, d'efficacité dont il faut que les enfants apprennent, pas toujours à effectuer eux-mêmes, mais comment elles sont faites et quelle confiance on peut avoir en elles est considérable : cela revient à leur apprendre ce qui est vraiment quantifiable et à le distinguer de ce qui ne l'est pas vraiment, la mesure tient un grande place dans la culture générale que nous voulons donner à nos enfants, les chiffres qui ne veulent rien dire ou qui sont carrément mensongers sont légion et une grande part de la littérature est consacrée à la mesure de choses telle la beauté, l'amour, le bonheur qui ne sont pas vraiment quantifiables et dont la mesure est éminemment subjective.
Un des intérêts majeurs de l'algorithmique est qu'elle s'insère entre les mathématiques, l'étude raisonnée des nombres et des objets géométriques et le reste du monde et donc faire de l'algorithmique c'est réfléchir à la, place des mathématique dans la vie et dans la société, modèle dont le caractère miraculeux a été souventes fois souligné de nombre de phénomènes naturels, physiques,

chimiques désormais aussi biologiques, la mathématique n'explique pas tout, loin s'en faut .
L'autre intérêt est que l'algorithmique a un champ d'action beaucoup plus vaste que les mathématiques et en particulier joue un rôle essentiel en linguistique et partant dans l'apprentissage des langues. La grammaire est une collection d'algorithmes : « trois cents tonnes de fraise sont vendues tous les jours au marché de Rungis », peut se dire « on vend tous les jours trois cents tonnes de fraise au marché de Rungis », ou encore « il se vend trois cents tonnes de fraises tous les jours au marché de Rungis » (voir grammaire Larousse). Le but d'une grammaire quelle qu'elle soit est de dire quelles phrases peuvent ainsi être susceptibles de trois formes formellement distinctes et rigoureusement synonymes. Plutôt que d'algorithme les grammaires parlent de règles, il y en a donc une qui dit que la première phrase de forme personnelle, le sujet est bien les tonnes de fraise qui sont vendues est équivalente aux deux autres de forme impersonnelle puisque les sujets sont on et il qui ne désigne rien ni personne. Mais en fait la règle est un algorithme qui permet de transformer chacune de ces trois phrases en les deux autres.
Cet exemple, plutôt simple, choisi au hasard dans le grammaire Larousse standard est pourtant moins simple qu'il n'y paraît car « trois plats de viande sont ensuite amenés » se transformera en « on amène ensuite trois plats de viande » sans problème alors que « il s'amène ensuite trois plats de viande » est du parler populaire que, je pense, il faut considérer comme fautif et décourager les élèves d'écrire, bien que nous ayons utilisé la même règle. Les professeurs de français savent mieux combien il y a de petites règles dans la grammaire qui permettent de construire des phrases ou de les comprendre, et à quel point ces règles sont sensibles au contexte, au sens du verbe, deux verbes qui apparaissent fort semblables n'autorisant pas exactement les mêmes constructions.
Personnellement je suis persuadé qu'un peu d'esprit algorithmique, consistant à toujours préciser au mieux les conditions d'utilisation d'une règle, et décrivant cette règle plus précisément qu'elle ne l'est d'habitude, clarifierait beaucoup la grammaire et en rendrait donc l'apprentissage plus aisé.
L'algorithmique est une façon de considérer et de décrire des suites d'actions qui mènent à un résultat, les suites de transformations d'une donnée en une autre au moyen de règles autorisées , de détecter et souvent corriger des erreurs déjà faite, de faire réagir un processus, un système en marche, aux sollicitations de son environnement pour en assurer le bon fonctionnement, son champ est ainsi pratiquement illimité et peu d'activités humaines échappent à son emprise, comme nous le montre l'informatisation de plus en plus poussée de la plupart d'entre elles, y compris celles qui sont réputées non scientifique. Tout ce que les encyclopédistes appelaient art, que nous appelons tantôt toujours art (ou artisanat), tantôt technique, tantôt science est d'abord fondamentalement une collection d'algorithmes.

D'une pierre, deux coups

Jean-Alain Roddier, IA-IPR de mathématiques

Afin de faire percevoir de façon ludique ce qu'est un algorithme, nous vous proposons ici de développer un algorithme qui vous permettra à la fin d'écrire votre année de naissance en code binaire, c'est-à-dire uniquement avec des 1 et des 0. Cela fait ici d'une pierre deux coups c'est-à-dire que cet article va permettre au lecteur de comprendre parfaitement ce qu'est un algorithme mais aussi de percevoir ce qu'est la base du numérique : le codage binaire.

L'algorithme développé sur un exemple

Pour faire simple, il nous faut commencer par un exemple : prenons l'entier 2015.

Afin d'obtenir l'écriture de 2015 en binaire, nous effectuons ce que l'on appelle les divisions euclidiennes successives - jusqu'à « épuisement » - de 2015 par 2 :

On divise 2015 par 2 : 2015 = **1007** x 2 + **1**
On prend le quotient obtenu avant (**1007**) que l'on divise à son tour par 2 :
1007 = **503** x 2 + **1**
On prend le quotient obtenu avant (**503**) que l'on divise à son tour par 2 :
503 = **251** x 2 + **1**
On reproduit le processus : **251** = **125** x 2 + **1**
125 = **62** x 2 + **1**
62 = **31** x 2 + **0**
31 = **15** x 2 + **1**
15 = **7** x 2 + **1**
7 = **3** x 2 + **1**
3 = **1** x 2 + **1**
On poursuit le processus jusqu'à obtenir un quotient égal à 0 : **1 = 0 x 2 + 1**

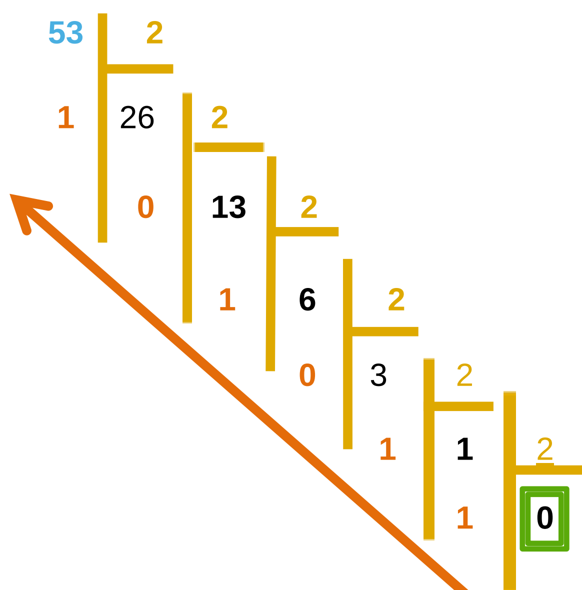
La liste des restes obtenus au fur et à mesure est la suivante : **1 1 1 1 1 0 1 1 1 1 1**.
Il suffit à présent de recopier cette liste à l'envers pour obtenir l'écriture en binaire de 2015, autrement dit : **1 1 1 1 1 0 1 1 1 1 1**.

Si l'on demande à présent ce que cela donne pour 2016, bon nombre de personnes ne vont pas vouloir recommencer ; peut-être parce que c'est une tâche assez rébarbative ou plus simplement parce que ces personnes craindront tout simplement de se tromper. De là à confier ce travail à un ordinateur, c'est le pas que nous allons franchir.

L'algorithme développé en langage naturel

Le processus que nous avons développé sur un exemple va être écrit de façon descriptive afin qu'il puisse être – si besoin est – transcrit en langage informatique (seul langage compréhensible par un ordinateur). Nous choisissons de développer cette partie sous la forme d'un scénario à proposer aux élèves dès le cycle 4. Ce scénario est basé sur une méthode consistant à s'appuyer sur des éléments acquis afin de « co-construire » des éléments nouveaux.

Au départ, nous proposons aux élèves de réaliser le codage binaire de 53. Prendre cet exemple très simple va permettre d'éviter « au maximum » les erreurs de calculs et faire en sorte que les élèves se sentent en confiance dès le début de la séance. L'ensemble de la classe est placée en situation de recherche pendant un temps suffisant afin que chaque élève arrive (avec « un peu » d'aide) à produire le code binaire attendu. Une fois cette phase de production réalisée, la correction est « co-contruite » avec tous les élèves, ce qui conduit à marquer au tableau de façon centrale les calculs successifs suivants :



53 divisé par 2 donne 26 et il reste 1
26 divisé par 2 donne 13 et il reste 0
13 divisé par 2 donne 6 et il reste 1
6 divisé par 2 donne 3 et il reste 0
3 divisé par 2 donne 1 et il reste 1
1 divisé par 2 donne 0 et il reste 1

On veille à repasser ces différents éléments en couleurs de manière à souligner leur rôle :

Repasser en bleu l'élément de départ
Repasser en ocre le processus répétitif
Entourer en vert la condition d'arrêt
Repasser en marron l'élément obtenu à l'arrivée.

On fait le compte rendu ligne par ligne des divisions posées, cela permet d'inscrire l'écriture de l'algorithme dans une présentation sous forme de lignes d'instructions.

Il est utile à ce stade de donner du sens à ce qui est produit, le fait d'écrire 53 sous la forme 110101 ne présente aucun intérêt pour les élèves sauf si on les invite à réfléchir à la notion de codage. Une question du type : « ***À quoi cela sert-il d'écrire 53 sous la forme 110101 ?*** » peut leur être judicieusement posée. On fera ainsi percevoir aux élèves cette bijection qui à un entier fait correspondre un unique code binaire et qui – réciproquement – à un code binaire fait correspondre un unique

entier. Coder, c'est en résumé, remplacer une information par une autre qui lui est parfaitement équivalente.

Il s'ensuit le remplissage de la colonne de gauche du tableau ci-dessous qui conduit à expliciter l'algorithme mis en œuvre dans le traitement de l'exemple. Les couleurs utilisées précédemment sont à nouveau mise en œuvre afin de faire ressortir les différentes étapes de l'algorithme : élément de départ, boucle répétitive, condition d'arrêt de la boucle, élément obtenu à l'arrivée.

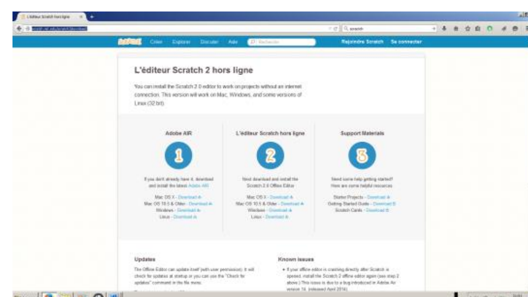
Langage naturel	Exemple
On prend un entier naturel non nul	**53**
On effectue les divisions successives par 2	**53 divisé par 2** donne 26 et il reste **1** 26 **divisé par 2** donne 13 et il reste **0** 13 **divisé par 2** donne 6 et il reste **1** 6 **divisé par 2** donne 3 et il reste **0** 3 **divisé par 2** donne 1 et il reste **1** 1 **divisé par 2** donne 0 et il reste **1**
Jusqu'à obtenir un quotient égal à zéro	
On construit la liste des restes dans l'ordre où on les a obtenus.	**101011**
Le résultat est obtenu en lisant la liste ainsi obtenue dans l'ordre inverse.	**110101**

L'algorithme développé en langage informatique

Il va s'agir à présent de transférer dans la machine l'algorithme que nous venons d'écrire en langage naturel ; pour ce faire, nous devons développer l'algorithme en langage de programmation. Nous allons dans la suite de cet article effectuer ce développement en utilisant deux logiciels libres à vertus pédagogiques qui ont déjà largement fait leur preuve :

- ❖ Scratch : ce logiciel Scratch est très adapté aux élèves débutants, son caractère jeu de LÉGO permettant de créer de l'appétence pour l'algorithmique ;
- ❖ Python : le Logiciel Python est plus « complet » ; il est destiné à des élèves davantage initiés à l'algorithmique.

Avec Scratch 



Le logiciel Scratch est téléchargeable en version à usage hors ligne à l'adresse suivante : http://scratch.mit.edu/scratch2download/.

Le développement de notre algorithme avec

Scratch peut être réalisé en « co-construction » avec les élèves en leur proposant de remplir deux tableaux (ci-dessous) dans lesquels on retrouve dans la colonne de droite les parties de notre algorithme écrit en langage naturel et dans la colonne de gauche l'algorithme développé en « langage Scratch ».

Développement de la partie concernant les divisions successives par 2

On crée une variable que l'on nomme « Nombre » et une liste que l'on nomme « Liste ».
La variable Nombre va contenir les dividendes successifs ; la liste « Liste » va quant à elle contenir les restes successifs.

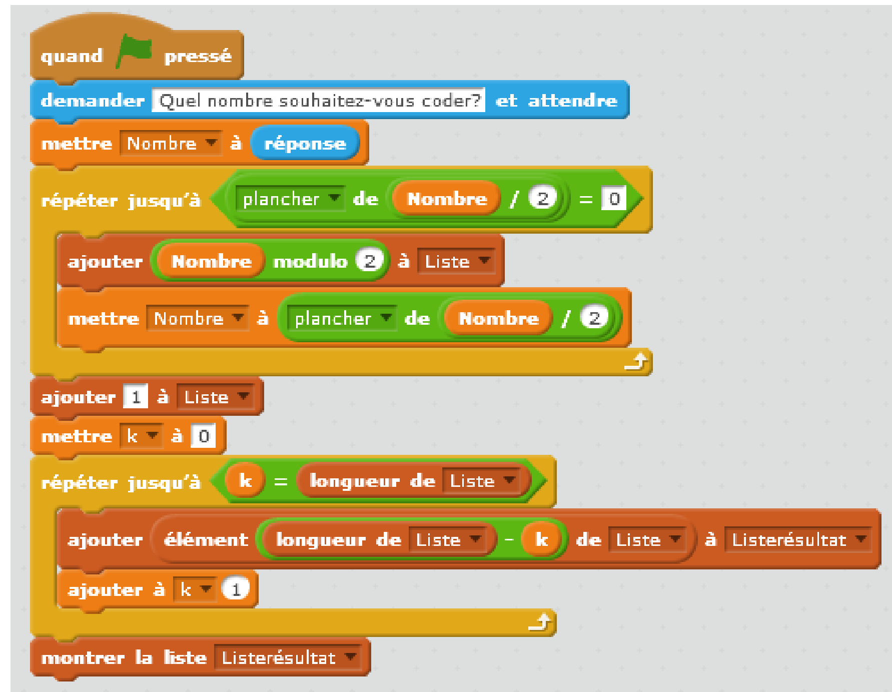
Langage Scratch	**Langage naturel**
quand [drapeau] pressé demander Quel nombre souhaitez-vous coder? et attendre mettre Nombre à réponse répéter jusqu'à plancher de Nombre / 2 = 0 ajouter Nombre modulo 2 à Liste mettre Nombre à plancher de Nombre / 2 ajouter 1 à Liste	On prend un entier naturel non nul On effectue les divisions successives par 2 **Jusqu'à obtenir un quotient égal à zéro** On construit la liste des restes dans l'ordre où on les a obtenus. La boucle n'étant pas parcourue lorsque le quotient est nul, il nous faut rajouter 1 en fin de liste.

Développement de la partie concernant la liste à lire en sens inverse

On crée une variable « k » et une nouvelle liste que l'on nomme « Listerésultat ». La variable « k » va permettre de parcourir la liste que nous avons remplie précédemment et la liste « Listerésultat » va contenir progressivement les éléments de cette liste lus dans l'ordre inverse.

Langage Scratch	Langage naturel
ajouter 1 à Liste mettre k à 0 répéter jusqu'à k = longueur de Liste ajouter élément longueur de Liste - k de Liste à Listerésultat ajouter à k 1 montrer la liste Listerésultat	**Le résultat est obtenu en lisant la liste ainsi obtenue dans l'ordre inverse.**

Il suffit à présent d'accoler les deux parties que nous venons de construire afin d'obtenir un programme qui code n'importe quel entier naturel non nul.
On pourra évoquer rapidement le codage de zéro qui est codé par « 0 » et dont le codage ne relève pas de cet algorithme.





Si l'on souhaite faire tourner le programme sur plusieurs exemples, il peut être utile de développer le petit utilitaire ci-contre destiné à effacer les deux listes avant de relancer le programme.

Pour lancer le programme, il suffit de cliquer sur le drapeau vert dans la fenêtre de gauche et de fournir - à l'invite de l'ordinateur - l'entier naturel non nul que l'on souhaite coder. En essayant avec mon année de naissance, nous avons obtenu : 11110101100. Qu'en est-il pour la vôtre ?

Avec Python 

Le logiciel de programmation Python est téléchargeable à l'adresse suivante : https://www.python.org/downloads/




```
>>> def base2(N):
        liste=[]
        while N//2!=0:
                liste.append(N%2)
                N=N//2
        liste.append(1)
        liste.reverse()
        print(liste)
```

Une fois le logiciel lancé, il suffit de taper dans la fenêtre active le programme à l'identique des lignes de codes informatiques données ci-contre.
Pour lancer le programme, vous pouvez – si vous êtes né(e) en 2003 - taper la suite d'instruction suivante : base2(*2003*).

Nous ne voudrions pas laisser penser ici que le passage de l'algorithme écrit en langage naturel aux lignes de codes informatiques soit automatique, ce n'est - en effet - pas le cas et le langage de programmation doit être parfaitement maitrisé afin que l'on soit certain d'avoir à l'aboutissement du processus de développement de l'algorithme, un programme qui fonctionne pour n'importe quelle valeur fournie en entrée. La programmation ne peut souffrir de quelconques failles, en le disant nous pensons à l'informatique développée dans les domaines à hauts risques humains : contrôle de centrales nucléaires ; logiciels intervenant en médecine, informatique embarquée dans l'aviation civile, … .